DEPARTMENT OF COMPUTER SCIENCE
COLLEGE OF SCIENCES
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23529-0247

HIGH-PERFORMANCE MONITORING ARCHITECTURE FOR LARGE-SCALE DISTRIBUTED SYSTEMS USING EVENT FILTERING

By
Dr. Kurt Maly, Principal Investigator

Supplemental Report

Prepared for
National Aeronautics and Space Administration
Langley Research Center
Attn: Joseph Murray, Mail Stop 128
Hampton, VA 23681-0001

Under
Grant# NAG-1-908
Wayne H. Bryant, Technical Monitor
Systems Integration Branch
ODURF #187264

Submitted by the
Old Dominion University Research Foundation
P.O. Box 6369
Norfolk, VA 23508-0369

May 1997

High-performance Monitoring Architecture for Large-scale Distributed Systems Using Event Filtering

Ehab Al-Shaer, Hussein Abdel-Wahab and Kurt Maly
Department of Computer Science
Old Dominion University
Norfolk, VA
(ehab,wahab,maly)@cs.odu.edu

Abstract

Monitoring is an essential process to observe and improve the reliability and the performance of large-scale distributed (LSD) systems. In an LSD environment, a large number of events is generated by the system components during its execution or interaction with external objects (e.g. users or processes). Monitoring such events is necessary for observing the run-time behavior of LSD systems and providing status information required for debugging, tuning and managing such applications. However, correlated events are generated
concurrently and could be distributed in various locations in the applications environment which complicates the management decisions process and thereby makes monitoring LSD systems an intricate task. We are developing a scalable high-performance monitoring architecture for LSD systems using an efficient event filtering mechanism to detect and classify interesting local and global events and disseminate the monitoring information to the corresponding end- points management applications (such as debugging and reactive control tools). Our architecture also supports dynamic and flexible reconfiguration of the monitoring mechanism via its instrumentation and subscription components. The intrusiveness of the monitoring process is minimized by reducing the event traffic and distributing the monitoring load.
In this short report, we describe and motivate the monitoring approach and application. We also refer to our publications for detailed explanation of the monitoring system components that we are developing to observe the run-time behavior of LSD systems and improve their reliability and performance.

1. Introduction and Motivation

As the Internet and Intranets technology advances, the demand of large-scale distributed (LSD) systems increases. Examples of LSD systems include large-scale collaborative distance learning, video conferencing, group-ware editing, distributed transaction systems and distributed interactive simulation. The LSD systems involve a large number of users or application entities, which are geographically dispersed over interconnected LANs (i.e. Intranets) or over WAN (i.e. Internet).
Due to the distributed nature and the large number of interactions (by participants or application entities) in LSD systems, *reliability* and *performance* of such applications become critical issues.
The LSD system developers or managers (could be human or software components) require feedback information on the system behavior for testing and debugging purposes or for fault recovery and performance tuning procedures.

Monitoring LSD systems is an effective means to observe applications at run-time and provide this feedback information to the system developers and system managers in order to improve reliability, robustness and performance of LSD systems.
However, in LSD systems, monitoring is much more complex because correlated events can be concurrent and distributed in the application environment. So the monitoring architecture is not only to detect and collect events generated by LSD systems but it also has to
check for any interesting correlated event (global event) occurred in the system.

A survey of monitoring and event filtering related work could be found in [3]. We classified the monitoring distributed systems related work into three classes: *hardware monitoring*, *software monitoring*, and *hybrid monitoring*. Examples of event filtering mechanisms in communication protocols are packet filters in communication protocols and distributed systems, and triggers in active databases.
The primary motivations for our work are (1) the necessity of employing an efficient monitoring technique for LSD systems to improve the quality and manageability of such applications, and (2) the lack of an existing monitoring technique that could meet our design goals of designing a scalable, high-performance, dynamic, flexible and non-intrusive monitoring architecture for LSD systems.

2. Work Objectives

Our main objective in this working designing a dynamic monitoring architecture for LSD systems by classifying primitive and composite events generated by distributed applications during their execution. The following is highlighting of the main design objectives:

• **Supporting An Efficient Monitoring Architecture**: Our monitoring uses an efficient filtering event filtering mechanism to classify and detect generated events and to reduce the large volume of event traffic that may be generated by LSD application and thereby minimizes the monitoring overhead (intrusiveness).

• **Supporting Dynamic Monitoring:** The consumer's subscriptions can be added, deleted and modified at run-time. This implies that the distributed systems can be monitored according to changes in the system behavior.

• **Minimizing Intervention:** The process of inserting the monitoring instructions inside the program body is called the *instrumentation* process. Our monitoring architecture must support:
(1) a simple technique to instrument the monitored programs with minimum interference from the developers or users, and
(2) a flexible instrumentation process to control the monitoring granularity and intrusiveness.

• **Minimizing the Monitoring Intrusiveness:** Our architecture must be used efficiently to monitor LSD systems without overwhelming the system resources such as the computation and the network resources.

• **Supporting Reactive Control Service:** The reactive control system enables the developers/users to define actions to be triggered, if
Certain events are detected. We call this process *Action Service*.

• **Supporting Priority-based Monitoring:** In LSD system, event could have different priorities. For example, *error* events have more priority than *warning* events. Thus, monitoring these events based on their priority is important to provide an efficient monitoring architecture.

3. Current Status and Future Work

We completed the design architecture of the monitoring components which is published in number of publications (See references). The filtering engine component is already implemented and tested in different environments. We are currently implementing the other components of the monitoring system such as the instrumentation, subscription and dissemination components.
We have also designed some important optimization techniques to improve the performance and scalability of the monitoring system. In the following, we present a short description of the proposed optimization techniques.

• **Filtering Optimization:** *Multi-path DAG* and *parallel filtering* techniques to enable an efficient filtering/monitoring mechanism, which increases the responsiveness of the monitoring system and reduces the intrusiveness in the monitoring environment.

• **Composition Optimization:** This type of optimization technique is used to increase the scalability of the monitoring system by employing an efficient organization of the monitoring information. In this case, the filtering or monitoring process does not increase linearly with the number of subscribers in the system. We proposed to techniques to optimize the filtering composition mechanism: *matching common predicates first* and *matching most-frequently-used predicates first*.

• **Space Requirement Optimization:** The former techniques are to optimize the computation requirement of the monitoring system. In this part of our research we proposed techniques to minimize the memory space required by the monitoring system such as tracking the event history.
Here, we list some of these techniques and for more details you may refer to [3,4].

References:

[1] Ehab Al-Shaer, Hussein Abdel-Wahab and Kurt Maly. "High-performance Monitoring Architecture for Large-scale Distributed Systems Using Event Filtering", *Third International Conference on Computer Science and Information*, 3:42-46, March 1997.

[2] Ehab Al-Shaer, "Event Filtering Framework: Key Criteria and Design Trade-offs", *21st IEEE Conference on Computer Software and Application*, August 1997.

[3] Ehab Al-Shaer, Hussein Abdel-Wahab and Kurt Maly, "High-performance Monitoring Architecture for Large-scale Distributed Systems Using Event Filtering", Ph.D. Proposal, TR-12-1997, Old Dominion University, Computer Science Department, Dec. 1996.

[4] Ehab Al-Shaer, "High-performance Event Filtering: Survey, Evaluation and Enhancements", Technical Report, TR-01-1997, Old Dominion University, Computer Science Department, July 1996.